

[illegible]

```
SSSSSSSS YY YY SSSSSSS MM MM 000000 UU UU
SSSSSSSS YY YY SSSSSSS MM MM 000000 UU UU
SS SS YY YY SS SS MMMM MMMM 00 00 UU UU
SS SS YY YY SS SS MMMM MMMM 00 00 UU UU
SS SS YY YY SS SS MM MM 00 00 UU UU
SSSSSS YY YY SSSSSS MM MM 00 00 UU UU
SSSSSS YY YY SSSSSS MM MM 00 00 UU UU
SS SS SS SS SS MM MM 00 00 UU UU
SS SS SS SS SS MM MM 00 00 UU UU
SSSSSS YY YY SSSSSSS MM MM 00 00 UU UU
SSSSSS YY YY SSSSSSS MM MM 000000 UUUUUUUUUU
SSSSSS YY YY SSSSSSS MM MM 000000 UUUUUUUUUU
.....
.....
.....
.....

LL LL IIIIII SSSSSSS
LL LL IIIIII SSSSSSS
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LL LL II
LLLLLLLLLL IIIIII SSSSSSS
LLLLLLLLLL IIIIII SSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
```

```
1 0001 0 MODULE SYSMOU (  
2 0002 0     LANGUAGE (BLISS32),  
3 0003 0     IDENT = 'V04-000'  
4 0004 0 ) =  
5 0005 1 BEGIN  
6 0006 1  
7 0007 1  
8 0008 1  
9 0009 1  
10 0010 1 *  
11 0011 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY  
12 0012 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.  
13 0013 1 *  ALL RIGHTS RESERVED.  
14 0014 1 *  
15 0015 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
16 0016 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
17 0017 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
18 0018 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
19 0019 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
20 0020 1 *  TRANSFERRED.  
21 0021 1 *  
22 0022 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
23 0023 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
24 0024 1 *  CORPORATION.  
25 0025 1 *  
26 0026 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
27 0027 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
28 0028 1 *  
29 0029 1 *****  
30 0030 1  
31 0031 1 ++  
32 0032 1  
33 0033 1 FACILITY: MOUNT Utility Structure Levels 1 & 2  
34 0034 1  
35 0035 1 ABSTRACT:  
36 0036 1  
37 0037 1     This module contains the code and data needed to mount the system  
38 0038 1     disk during system initialization.  
39 0039 1  
40 0040 1 ENVIRONMENT:  
41 0041 1  
42 0042 1     STARLET operating system, including privileged system services  
43 0043 1     and internal exec routines.  
44 0044 1  
45 0045 1 --  
46 0046 1  
47 0047 1  
48 0048 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 1-Nov-1977 19:02  
49 0049 1  
50 0050 1 MODIFIED BY:  
51 0051 1  
52 0052 1     V03-010 CDS0005      Christian D. Saether      29-Aug-1984  
53 0053 1     Use STAND_ALONE_REBUILD routine to avoid unnecessary  
54 0054 1     rebuilds.  
55 0055 1  
56 0056 1     V03-009 CDS0004      Christian D. Saether      2-Aug-1984  
57 0057 1     Test the sysgen flag REBLDSYSD to determine whether
```

58	0058	1	rebuild should be performed.
59	0059	1	
60	0060	1	V03-008 HH0041 Hai Huang 24-Jul-1984
61	0061	1	Remove REQUIRE 'OBJDS:[VMSLIB.OBJ]MOUNTMSG.REQ'.
62	0062	1	
63	0063	1	V03-007 HH0018 Hai Huang 06-May-1984
64	0064	1	Use \$GETDVI to obtain the physical device name of the
65	0065	1	system device.
66	0066	1	
67	0067	1	V03-006 TMH0006 Tim Halvorsen 14-Apr-1984
68	0068	1	Add MOUNT_FLAGS to list of dummy storage needed for
69	0069	1	linked-in-MOUNT.
70	0070	1	
71	0071	1	V03-005 CDS0003 Christian D. Saether 19-Oct-1983
72	0072	1	Now that volume rebuild works, allow FID and EXT caching.
73	0073	1	
74	0074	1	V03-004 TCM0001 Trudy C. Matthews 19-Aug-1983
75	0075	1	Interlock mounts of the system disk with other potential
76	0076	1	mounters of the same disk in the cluster. Add cluster
77	0077	1	consistency checking routines.
78	0078	1	
79	0079	1	V03-003 CDS0002 Christian D. Saether 15-Aug-1983
80	0080	1	Set OPT_NOEXT_C, OPT_NOFID_C, OPT_NOQUO_C, and OPT_WTHRU
81	0081	1	to REALCY disable caching.
82	0082	1	
83	0083	1	V03-002 CDS0001 Christian D. Saether 5-Aug-1983
84	0084	1	Temporarily disable caching on system disk until
85	0085	1	xqp cluster rebuild works.
86	0086	1	
87	0087	1	V03-001 STJ3061 Steven T. Jeffreys, 04-Mar-1983
88	0088	1	Added definitions of DEVICE_INDEX and CALLERS_ACMOD.
89	0089	1	These parallel definitions in VMOUNT.
90	0090	1	
91	0091	1	V02-008 STJ0202 Steven T. Jeffreys, 05-Feb-1982
92	0092	1	Make sure the OPT_MOUNTVER bit gets set. The first
93	0093	1	attempt at this ended in disaster.
94	0094	1	
95	0095	1	V02-007 STJ0175 Steven T. Jeffreys, 06-Jan-1982
96	0096	1	Set up the database to ensure the system disk
97	0097	1	is a candidate for mount verification.
98	0098	1	
99	0099	1	V02-006 ACG0248 Andrew C. Goldstein, 31-Dec-1981 16:56
100	0100	1	Use default logical name, fix use of \$GETDEV
101	0101	1	
102	0102	1	V02-005 ACG0181 Andrew C. Goldstein, 13-Oct-1980 15:37
103	0103	1	Fix cross facility references
104	0104	1	
105	0105	1	V0104 ACG0123 Andrew C. Goldstein, 12-Feb-1980 18:23
106	0106	1	Integrate disk rebuild into MOUNT
107	0107	1	
108	0108	1	V0103 ACG0079 Andrew C. Goldstein, 11-Nov-1979 19:32
109	0109	1	MOUNT changes for write-back caching
110	0110	1	
111	0111	1	V0102 ACG0072 Andrew C. Goldstein, 22-Oct-1979 13:53
112	0112	1	Check primary and secondary device char
113	0113	1	
114	0114	1	V101 ACG0003 Andrew C. Goldstein, 28-Dec-1978 15:23

SYSMOU
V04-000

J 4
16-Sep-1984 02:12:43
14-Sep-1984 13:16:57

VAX-11 Bliss-32 V4.0-742
[SYSINI.SRC]SYSMOU.B32;1

Page 3
(1)

```

: 115      0115 1 |      Add global variables for multi-volume MOUNT
: 116      0116 1 |
: 117      0117 1 |
: 118      0118 1 |      V100      ACG0001      Andrew C. Goldstein, 28-Dec-1978 15:22
: 119      0119 1 |      Previous revision history moved to SYSINIT.REV
: 120      0120 1 |      **
: 121      0121 1 |
: 122      0122 1 |      LIBRARY 'SYSS$LIBRARY:LIB.L32';
: 123      0123 1 |      REQUIRE 'LIB$:MOUDEF.B32';
: 124      0655 1 |
: 125      0656 1 |
: 126      0657 1 |      FORWARD ROUTINE
: 127      0658 1 |      MOUNT_SYSTEM,      | main routine
: 128      0659 1 |      MAIN_HANDLER;      | condition handler

```

```
130 0660 1 1+
131 0661 1
132 0662 1 Own storage for general use in the MOUNT utility
133 0663 1
134 0664 1 1-
135 0665 1
136 0666 1 GLOBAL
137 0667 1
138 0668 1 STORED_CONTEXT : BITVECTOR [32], ! store the context of some 1 time only
139 0669 1 MOUNT_FLAGS : LONG INITIAL(0), ! MOUNT flags
140 0670 1 LOCK_STATUS : VECTOR [2], ! Lock status block for $ENQ call
141 0671 1 DEVICE_INDEX : LONG, ! index into PHYS_NAME array
142 0672 1 CALLERS_ACMOD : LONG, ! caller's access mode
143 0673 1 CLEANUP_FLAGS : BITVECTOR [32], ! error cleanup status flags
144 0674 1 CHANNEL : ! channel number for I/O
145 0675 1 MAILBOX_CHANNEL : ! channel number of ACP termination mailbox
146 0676 1 PHYS_BUFFER : VECTOR [20, BYTE], ! buffer to construct phys device name
147 0677 1
148 0678 1 PHYS_NAME : VECTOR [2]
149 0679 1 INITIAL (0, PHYS_BUFFER), ! descriptor of physical device name
150 0680 1 LOG_BUFFER : VECTOR [20, BYTE], ! buffer to construct logical name
151 0681 1 HOME_BLOCK : BBLOCK [512], ! buffer for volume header label or home block
152 0682 1 DEVICE_CHAR : BBLOCK [DIB$K_LENGTH], ! buffer for device characteristics
153 0683 1
154 0684 1 DEVICE_CHAR2 : BBLOCK [DIB$K_LENGTH], ! buffer for 2nd device characteristics
155 0685 1
156 0686 1 HOMEBLOCK_LBN, ! LBN of home block read
157 0687 1 HEADER_LBN, ! LBN of file header
158 0688 1 DEV_INDEX : ! index into device data table
159 0689 1 USER_STATUS : VECTOR [2], ! status return for various routines
160 0690 1 CURRENT_RVN, ! RVN of volume being mounted
161 0691 1 CURRENT_VCB : REF BBLOCK, ! address of VCB used by CHECK_HEADER2
162 0692 1 REAL_RVT : REF BBLOCK, ! address of RVT allocated for volume set
163 0693 1 REAL_VCB : REF BBLOCK, ! address of VCB allocated for volume
164 0694 1 REAL_FCB : REF BBLOCK, ! address of FCB allocated for volume
165 0695 1 REAL_WCB : REF BBLOCK, ! address of window allocated for volume
166 0696 1 REAL_VCA : REF BBLOCK, ! address of cache block allocated
167 0697 1 REAL_AQB : REF BBLOCK, ! address of AQB allocated for volume
168 0698 1 LOG_ENTRY : REF BBLOCK, ! address of logical name entry
169 0699 1 MTL_ENTRY : REF BBLOCK, ! address of mounted volume list entry
170 0700 1 SLOG_ENTRY : REF BBLOCK, ! address of volume set logical name entry
171 0701 1 SMTL_ENTRY : REF BBLOCK, ! address of volume set mounted volume list entry
172 0702 1
173 0703 1 DEVCHAR_DESC : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR), ! descriptor for device characteristics
174 0704 1
175 0705 1 DEVCHAR_DESC2 : VECTOR [2] INITIAL (DIB$K_LENGTH, DEVICE_CHAR2); ! descriptor for device characteristics
176 0706 1
177 0707 1
178 0708 1
179 0709 1
180 0710 1 1+
181 0711 1
182 0712 1 The following area is a hand crafted mount parser output suitable for
183 0713 1 mounting the system disk.
184 0714 1
185 0715 1 1-
186 0716 1
```

```
: 187      0717 1 GLOBAL
: 188      0718 1
: 189      0719 1 MOUNT_OPTIONS : BITVECTOR [64] ! option flags
: 190      0720 2 INITIAL (      ! First 32 bits
: 191      0721 2      1^OPT_SYSTEM OR
: 192      0722 2      1^OPT_WRITE OR
: 193      0723 2      1^OPT_BLOCK OR
: 194      0724 2      1^OPT_OVR_ID OR
: 195      0725 2      1^OPT_DEVICE OR
: 196      0726 2      1^OPT_LABEL)
: 197      0727 2      (1^(OPT_MOUNTVER-32) OR      ! Last 32 bits
: 198      0728 2      1^(OPT_NOQUO_C-32) OR
: 199      0729 2      1^(OPT_WTHRU-32))
: 200      0730 1
: 201      0731 1 PROTECTION : INITIAL (0), ! value of /PROTECTION switch
: 202      0732 1 OWNER_UIC : INITIAL (0), ! value of /OWNER_UIC switch
: 203      0733 1 USER_OIC : INITIAL (0), ! value of /USER_OIC switch
: 204      0734 1 EXTENSION : INITIAL (0), ! value of /EXTENSION switch
: 205      0735 1 WINDOW : INITIAL (0), ! value of /WINDOW switch
: 206      0736 1 ACCESSED : INITIAL (0), ! value of /ACCESSED switch
: 207      0737 1 BLOCKSIZE : INITIAL (0), ! value of /BLOCK switch
: 208      0738 1 EXT_CACHE : INITIAL (0), ! value of /CACHE=(EXTENT=n) switch
: 209      0739 1 FID_CACHE : INITIAL (0), ! value of /CACHE=(FILE=n) switch
: 210      0740 1 QUO_CACHE : INITIAL (0), ! value of /CACHE=(QUOTA=n) switch
: 211      0741 1 EXT_LIMIT : INITIAL (0), ! value of /CACHE=(LIMIT=n) switch
: 212      0742 1 DEVICE_COUNT : INITIAL (1), ! number of devices specified
: 213      0743 1 LABEL_COUNT : INITIAL (1), ! number of volume labels specified
: 214      0744 1 LOG_NAME : VECTOR [2], ! logical name of system disk
: 215      0745 1 STRUCT_NAME : VECTOR [2], ! descriptor of structure name
: 216      0746 1 VID_STRING : VECTOR [2], ! descriptor of VISUAL_ID string
: 217      0747 1 COMMENT_STRING : VECTOR [2], ! descriptor of COMMENT string
: 218      0748 1 ACP_STRING : VECTOR [2], ! descriptor of ACP device or name string
: 219      0749 1 DRIVE_COUNT : VECTOR [1], ! value of /DRIVES switch
: 220      0750 1
: 221      0751 1 PARSE_IMP_END : VECTOR [0]; ! end of data area
: 222      0752 1
: 223      0753 1 GLOBAL BIND
: 224      0754 1 LABEL_STRING = DESCRIPTOR ('SYSTEMDISK') : VECTOR;
: 225      0755 1 ! dummy volume label of system disk
```

```
227 0756 1 GLOBAL ROUTINE MOUNT_SYSTEM (SYS_CHANNEL) =
228 0757 1
229 0758 1 ++
230 0759 1
231 0760 1 FUNCTIONAL DESCRIPTION:
232 0761 1
233 0762 1 This routine mounts the system disk (i.e., the disk to which the
234 0763 1 channel is assigned) and starts the ACP.
235 0764 1
236 0765 1
237 0766 1 CALLING SEQUENCE:
238 0767 1 MOUNT_SYSTEM (ARG1)
239 0768 1
240 0769 1 INPUT PARAMETERS:
241 0770 1 ARG1: channel number assigned to disk
242 0771 1
243 0772 1 IMPLICIT INPUTS:
244 0773 1 own storage of this module
245 0774 1
246 0775 1 OUTPUT PARAMETERS:
247 0776 1 NONE
248 0777 1
249 0778 1 IMPLICIT OUTPUTS:
250 0779 1 NONE
251 0780 1
252 0781 1 ROUTINE VALUE:
253 0782 1 1 if successful, assorted statuses if not
254 0783 1
255 0784 1 SIDE EFFECTS:
256 0785 1 system disk mounted, ACP started, logical name created
257 0786 1
258 0787 1 --
259 0788 1
260 0789 2 BEGIN
261 0790 2
262 0791 2 LOCAL
263 0792 2 MOUNT_IOSB : VECTOR [2],
264 0793 2 ALLDEVNAM_BUF : VECTOR [NAMEBUF_LEN, BYTE]
265 0794 2 INITIAL (BYTE ('MOUS', REP NAMEBUF_LEN-4 OF (' '))),
266 0795 2 ALLDEVNAM_DESC : VECTOR [2] INITIAL (0, ALLDEVNAM_BUF),
267 0796 2 DEVICE_ITMLST : BBLOCK [(2 * 12) + 4] INITIAL
268 0797 2
269 0798 2
270 0799 2 1st item - device name
271 0800 2
272 0801 2 (WORD (20), : Length of dev name buffer
273 0802 2 WORD (DVIS_DEVNAM), : Item code for device name
274 0803 2 LONG (PHYS_BUFFER), : Dev name buffer address
275 0804 2 LONG (PHYS_NAME), : Returned dev name length
276 0805 2
277 0806 2 2nd item - allocation class name
278 0807 2
279 0808 2 WORD (NAMEBUF_LEN - 4),
280 0809 2 WORD (DVIS_ALLDEVNAM),
281 0810 2 LONG (ALLDEVNAM_BUF + 4),
282 0811 2 LONG (ALLDEVNAM_DESC),
283 0812 2
```

```
284      0813      ! End of list.
285      0814      !
286      0815      ! LONG (0)),
287      0816      STATUS,      ! system service status
288      0817      P;           ! pointer into characteristics block
289      0818
290      0819      EXTERNAL
291      0820      EXESGL_STATIC_FLAGS : ADDRESSING MODE (GENERAL) BITVECTOR [32],
292      0821      DEV_CTR      : BBLOCK FIELD (DC);
293      0822      ! device value block context fields
294      0823
295      0824      EXTERNAL LITERAL
296      0825      EXESV_REBLDSYSD;
297      0826
298      0827      EXTERNAL ROUTINE
299      0828      READ_HOMEBLOCK,      ! read disk home block
300      0829      MOUNT_DISK1,      ! mount disk, level 1
301      0830      MOUNT_DISK2,      ! mount disk, level 2
302      0831      STAND_ALONE_REBUILD, ! rebuild disk bitmaps and quota file
303      0832      GET_DEVICE_CONTEXT; ! get device lock value block context
304      0833
305      0834
306      0835      ! Enable the condition handler.
307      0836      !
308      0837
309      0838      ENABLE MAIN_HANDLER;
310      0839
311      0840      CALLERS_ACMOD = PSLSC_SUPER;      ! used for logical name access mode
312      0841      CHANNEL = .SYS_CHANNEL;
313      0842
314      0843      !
315      0844      ! Take out a lock to synchronize all mounts of this device in a cluster.
316      0845      ! First we must construct the lock resource name (use the allocation class
317      0846      ! name returned by $GETDVI).
318      0847      !
319      0848
320      0849      STATUS = $GETDVIW (CHAN = .CHANNEL,
321      0850      ITMLST = DEVICE_ITMLST,
322      0851      EFN = MOUNT_EFN,
323      0852      IOSB = MOUNT_IOSB);
324      0853      IF NOT .STATUS THEN ERR_EXIT (.STATUS);
325      0854      ALLDEVNAM_DESC[0] = .ALLDEVNAM_DESC[0] + 4;
326      0855
327      0856      STATUS = $ENQW (LKMODE = LCKSK_EXMODE,
328      0857      LKSB = LOCK_STATUS,
329      0858      FLAGS = LCKSM_SYSTEM,
330      0859      RESNAM = ALLDEVNAM_DESC,
331      0860      EFN = MOUNT_EFN,
332      0861      ACMODE = PSLSC_EXEC);
333      0862      IF NOT .STATUS THEN ERR_EXIT (.STATUS);
334      0863
335      0864      ! Get the device characteristics and do device type validation: Make sure
336      0865      ! the device is mountable at all, and check that the mount qualifiers are
337      0866      ! consistent with the device type. A mismatch between primary and secondary
338      0867      ! device characteristics indicates a spooled device or something else strange.
339      0868      ! Reject such.
340      0869
```

```
341 0870 2
342 0871 22 $GETCHN (CHAN = .CHANNEL, PRIBUF = DEVCHAR_DESC, SCDBUF = DEVCHAR_DESC2);
343 0872 22
344 0873 22 IF CH$NEQ (DIB$K_LENGTH, DEVICE_CHAR, DIB$K_LENGTH, DEVICE_CHAR2, 0)
345 0874 22 OR NOT .DEVICE_CHAR[DEV$V_FOD]
346 0875 22 THEN ERR_EXIT (SS$_NOTFILEDEV);
347 0876 22
348 0877 22 IF NOT .DEVICE_CHAR[DEV$V_AVL]
349 0878 22 THEN ERR_EXIT (SS$_DEVOFFLINE);
350 0879 22
351 0880 22 IF .DEVICE_CHAR[DEV$V_MNT]
352 0881 22 THEN ERR_EXIT (SS$_DEVMOUNT);
353 0882 22
354 0883 22 IF .DEVICE_CHAR[DEV$V_SQD]
355 0884 22 THEN ERR_EXIT (SS$_NOTFILEDEV);
356 0885 22
357 0886 22
358 0887 22 ! The following is for reference only. The physical device name is now
359 0888 22 obtained with the $GETDVIW system service, rather than formatting device
360 0889 22 name and the unit number.
361 0890 22
362 0891 22 ! Construct the physical device name by appending the ascii unit number to
363 0892 22 the device name in the device characteristics.
364 0893 22
365 0894 22
366 0895 22 PHYS_NAME[0] = 20;
367 0896 22 PHYS_NAME[1] = PHYS_BUFFER;
368 0897 22 $FAO (
369 0898 22     DESCRIPTOR ('_!AC!UW:'),
370 0899 22     PHYS_NAME[0],
371 0900 22     PHYS_NAME[0],
372 0901 22     DEVICE_CHAR + .DEVICE_CHAR[DIB$W_DEVNAMOFF],
373 0902 22     .DEVICE_CHAR[DIB$W_UNIT]
374 0903 22 );
375 0904 22
376 0905 22 ! Now attempt to read the home block or volume header label, as appropriate
377 0906 22 for the device type.
378 0907 22
379 0908 22
380 0909 22 STATUS = READ_HOMEBLOCK (LABEL_STRING[0]);
381 0910 22
382 0911 22 MOUNT_OPTIONS[OPT_IS_FILES11] = 1;      ! assume volume is Files-11
383 0912 22 IF NOT .STATUS
384 0913 22 AND .STATUS NEQ SS$ INCVOLLABEL
385 0914 22 THEN ERR_EXIT (.STATUS);
386 0915 22
387 0916 22 IF NOT (STATUS = KERNEL_CALL (GET_DEVICE_CONTEXT))
388 0917 22 THEN ERR_EXIT (.STATUS);
389 0918 22
390 0919 22 IF .MOUNT_OPTIONS[OPT_IS_FILES11B]
391 0920 22 THEN MOUNT_DISK2 ()
392 0921 22 ELSE MOUNT_DISK1 ();
393 0922 22
394 0923 22 ! Rebuild the volume if it was improperly dismounted.
395 0924 22
396 0925 22
397 0926 22 IF .CLEANUP_FLAGS[CLF_REBUILD]
```

```

398 0927 2 AND .EXESGL_STATIC_FLAGS [EXESV_REBLDSYSD]
399 0928 2 THEN
400 0929 2 BEGIN
401 0930 2 ERR_MESSAGE (MOUN$ REBUILD);
402 0931 2 STAND_ALONE_REBUILD (.CHANNEL);
403 0932 2 END;
404 0933 2
405 0934 2 IF .LOCK_STATUS [1] NEQ 0
406 0935 2 THEN
407 0936 2 BEGIN
408 0937 2 $DEQ (LKID = .LOCK_STATUS [1]);
409 0938 2 LOCK_STATUS [1] = 0;
410 0939 2 END;
411 0940 2
412 0941 2 1
413 0942 2 1 END;

```

! end of routine MOUNT_COMMAND

```

.TITLE SYSMOU
.IDENT \V04-000\

.PSECT $PLITS$,NOWRT,NOEXE,2

4B 53 49 44 4D 45 54 53 59 53 00000 P.AAB: .ASCII \SYSTEMDISK\
0000A .BLKB 2
0000C P.AAA: .LONG 10
00000000' 00010 .ADDRESS P.AAB
24 55 4F 4D 00014 P.AAC: .ASCII \MOUS\
20 00018 .ASCII \
20 00019 .ASCII \
20 0001A .ASCII \
20 0001B .ASCII \
20 0001C .ASCII \
20 0001D .ASCII \
20 0001E .ASCII \
20 0001F .ASCII \
20 00020 .ASCII \
20 00021 .ASCII \
20 00022 .ASCII \
20 00023 .ASCII \
20 00024 .ASCII \
20 00025 .ASCII \
20 00026 .ASCII \
20 00027 .ASCII \
20 00028 .ASCII \
20 00029 .ASCII \
20 0002A .ASCII \
20 0002B .ASCII \
20 0002C .ASCII \
20 0002D .ASCII \
20 0002E .ASCII \
20 0002F .ASCII \
20 00030 .ASCII \
20 00031 .ASCII \
20 00032 .ASCII \
20 00033 .ASCII \
0014 00034 P.AAD: .WORD 20

```

```
0020 00036 .WORD 32
00000000 00038 .ADDRESS PHYS_BUFFER
00000000 0003C .ADDRESS PHYS_NAME
001C 00040 .WORD 28
00EC 00042 .WORD 236
00000000 00044 .LONG 0
00000000 00048 .LONG 0
00000000 0004C .LONG 0

.PSECT $GLOBAL$,NOEXE,2

00000 STORED_CONTEXT::
      .BLKB 4
00000000 00004 MOUNT_FLAGS::
      .LONG 0
00008 LOCK_STATUS::
      .BLKB 8
00010 DEVICE_INDEX::
      .BLKB 4
00014 CALLERS_ACMOD::
      .BLKB 4
00018 CLEANUP_FLAGS::
      .BLKB 4
0001C CHANNEL::
      .BLKB 4
00020 MAILBOX_CHANNEL::
      .BLKB 4
00024 PHYS_BUFFER::
      .BLKB 20
00000000 00038 PHYS_NAME::
      .LONG 0
00000000 0003C .ADDRESS PHYS_BUFFER
00040 LOG_BUFFER::
      .BLKB 20
00054 HOME_BLOCK::
      .BLKB 512
00254 DEVICE_CHAR::
      .BLKB 116
002C8 DEVICE_CHAR2::
      .BLKB 116
0033C HOMEBLOCK_LBN::
      .BLKB 4
00340 HEADER_LBN::
      .BLKB 4
00344 DEV_INDEX::
      .BLKB 4
00348 USER_STATUS::
      .BLKB 8
00350 CURRENT_RVN::
      .BLKB 4
00354 CURRENT_VCB::
      .BLKB 4
00358 REAL_RVT::
      .BLKB 4
0035C REAL_VCB::
      .BLKB 4
00360 REAL_FCB::
```

E 5
16-Sep-1984 02:12:43
14-Sep-1984 13:16:57VAX-11 Bliss-32 V4.0-742
[SYSINI.SRC]SYSMOU.B32;1Page 11
(3)

			.BLKB	4	
	00364	REAL_WCB::			
			.BLKB	4	
	00368	REAL_VCA::			
			.BLKB	4	
	0036C	REAL_AQB::			
			.BLKB	4	
	00370	LOG_ENTRY::			
			.BLKB	4	
	00374	MTL_ENTRY::			
			.BLKB	4	
	00378	SLOG_ENTRY::			
			.BLKB	4	
	0037C	SMTL_ENTRY::			
			.BLKB	4	
	00000074	00380 DEVCHAR_DESC::			
			.LONG	116	
	00000000'	00384	ADDRESS	DEVICE_CHAR	:
	00000074	00388 DEVCHAR_DESC2::			
			.LONG	116	:
	00000000'	0038C	ADDRESS	DEVICE_CHAR2	:
00424000	C0408300	00390 MOUNT_OPTIONS::			:
			.LONG	-1069513984, 4341760	:
	00000000	00398 PROTECTION::			:
			.LONG	0	:
	00000000	0039C OWNER_UIC::			:
			.LONG	0	:
	00000000	003A0 USER_UIC::			:
			.LONG	0	:
	00000000	003A4 EXTENSION::			:
			.LONG	0	:
	00000000	003A8 WINDOW::			:
			.LONG	0	:
	00000000	003AC ACCESSED::			:
			.LONG	0	:
	00000000	003B0 BLOCKSIZE::			:
			.LONG	0	:
	00000000	003B4 EXT_CACHE::			:
			.LONG	0	:
	00000000	003B8 FID_CACHE::			:
			.LONG	0	:
	00000000	003BC QUO_CACHE::			:
			.LONG	0	:
	00000000	003C0 EXT_LIMIT::			:
			.LONG	0	:
	00000001	003C4 DEVICE_COUNT::			:
			.LONG	1	:
	00000001	003C8 LABEL_COUNT::			:
			.LONG	1	:
		003CC LOG_NAME::			:
			.BLKB	8	
		003D4 STRUCT_NAME::			
			.BLKB	8	
		003DC VID_STRING::			
			.BLKB	8	
		003E4 COMMENT_STRING::			
			.BLKB	8	
		003EC ACP_STRING::			

.BLKB 8
003F4 DRIVE_COUNT::
.BLKB 4
003FB PARSE_IMP_END::
.BLKB 0

LABEL_STRING== P.AAA
.EXTRN EXESGL_STATIC_FLAGS
.EXTRN DEV_CTX, EXESV_REBLDSYSD
.EXTRN READ_HOMEBLOCK, MOUNT_DISK1
.EXTRN MOUNT_DISK2, STAND_ALONE_REBUILD
.EXTRN GET_DEVICE_CONTEXT
.EXTRN SYS\$GETDVIW, SYS\$ENQW
.EXTRN SYS\$GETCHN, SYS\$CMKRNL
.EXTRN SYS\$DEQ

.PSECT \$CODE\$,NOWRT,2

24	AE	0000'	57	00000000G	00	00FC	00000	.ENTRY	MOUNT_SYSTEM, Save R2,R3,R4,R5,R6,R7	0756
			56	0000'	CF	9E	00002	MOVAB	LIB\$STOP, R7	
			5E	B4	AE	9E	00009	MOVAB	DEVICE_CHAR, R6	
			CF		20	28	00012	MOVAB	-76(SPT, SP	
				1C	AE	D4	00019	MOVAB	#32, P.AAC, ALLDEVNAM_BUF	0794
				24	AE	9E	0001C	CLRL	ALLDEVNAM_DESC	
	6E	20	AE		1C	28	00021	MOVAB	ALLDEVNAM_BUF, ALLDEVNAM_DESC+4	
		10	CF		AE	9E	00027	MOVAB	#28, P.AAD, DEVICE_ITMLST	0815
		14	AE		AE	9E	0002C	MOVAB	ALLDEVNAM_BUF+4, DEVICE_ITMLST+16	0810
			6D	013D	CF	DE	00031	MOVAB	ALLDEVNAM_DESC, DEVICE_ITMLST+20	0794
		FDC0	C6		02	D0	00036	MOVAL	14\$, (FP)	0815
		FDC8	C6		7E	7C	00041	MOVL	#2, CALLERS_ACMOD	0840
				04	AC	D0	0003B	MOVL	SYS_CHANNEL, CHANNEL	0841
					7E	D4	00043	CLRL	-(SP)	0852
				50	AE	9F	00045	CLRL	-(SP)	
				10	AE	9F	00048	PUSHAB	MOUNT_IOSB	
					7E	D4	0004B	PUSHAB	DEVICE_ITMLST	
				FDC8	C6	DD	0004D	CLRL	-(SP)	
					1A	DD	00051	PUSHL	CHANNEL	
		00000000G	00		08	FB	00053	PUSHL	#26	
			54		50	D0	0005A	CALLS	#8, SYS\$GETDVIW	
			05		54	E8	0005D	MOVL	R0, STATUS	
					54	DD	00060	BLBS	STATUS, 1\$	0853
					01	FB	00062	PUSHL	STATUS	
			67		04	C0	00065	CALLS	#1, LIB\$STOP	
		1C	AE		01	7D	00069	ADDL2	#4, ALLDEVNAM_DESC	0854
			7E		7E	7C	0006C	MOVQ	#1, -(SP)	0861
					7E	7C	0006E	CLRL	-(SP)	
					AE	9F	00070	CLRL	-(SP)	
				34	10	DD	00073	PUSHAB	ALLDEVNAM_DESC	
					C6	9F	00075	PUSHL	#16	
				FDB4	05	DD	00079	PUSHAB	LOCK_STATUS	
					1A	DD	0007B	PUSHL	#5	
		00000000G	00		08	FB	0007D	PUSHL	#26	
			54		50	D0	00084	CALLS	#11, SYS\$ENQW	
			05		54	E8	00087	MOVL	R0, STATUS	
					54	DD	0008A	BLBS	STATUS, 2\$	0862
			67		01	FB	0008C	PUSHL	STATUS	
								CALLS	#1, LIB\$STOP	

74	A6	00000000G	00	0134	C6	9F	0008F	2%:	PUSHAB	DEVCHAR_DESC2	0871
			66		7E	D4	00093		CLRL	-(SP)	
	08	01	A6	012C	C6	9F	00095		PUSHAB	DEVCHAR_DESC	
			7E		7E	D4	00099		CLRL	-(SP)	
			67	FDC8	C6	DD	00098		PUSHL	CHANNEL	
			66		05	FB	0009F		CALLS	#5, SYSSGETCHN	
			67	0074	8F	29	000A6		CMPCS	#116, DEVICE_CHAR, DEVICE_CHAR2	0873
			67		05	12	000AD		BNEQ	3%	
			67		06	E0	000AF		BBS	#6, DEVICE_CHAR+1, 4%	0874
			67	01CC	8F	3C	000B4	3%:	MOVZWL	#460, -(SPT)	0875
			67		01	FB	000B9		CALLS	#1, LIB\$STOP	
			67		02	E0	000BC	4%:	BBS	#2, DEVICE_CHAR+2, 5%	0877
			67	84	8F	9A	000C1		MOVZBL	#132, -(SPT)	0878
			67		01	FB	000C5		CALLS	#1, LIB\$STOP	
			67		03	E1	000C8	5%:	BBC	#3, DEVICE_CHAR+2, 6%	0880
			67	6C	8F	9A	000CD		MOVZBL	#108, -(SPT)	0881
			67		01	FB	000D1		CALLS	#1, LIB\$STOP	
			67		05	E1	000D4	6%:	BBC	#5, DEVICE_CHAR, 7%	0883
			67	01CC	8F	3C	000D8		MOVZWL	#460, -(SPT)	0884
			67		01	FB	000DD		CALLS	#1, LIB\$STOP	
			67	0000'	CF	9F	000E0	7%:	PUSHAB	LABEL STRING	0909
			67		01	FB	000E4		CALLS	#1, READ HOMEBLOCK	
			67		50	D0	000E9		MOVL	R0, STATUS	
			67		02	88	000EC		BISB2	#2, MOUNT_OPTIONS+4	0911
			67		54	E8	000F1		BLBS	STATUS, 8%	0912
			67		54	D1	000F4		CMPL	STATUS, #268	0913
			67		05	13	000FB		BEQL	8%	
			67		54	DD	000FD		PUSHL	STATUS	0914
			67		01	FB	000FF		CALLS	#1, LIB\$STOP	
			67		7E	D4	00102	8%:	CLRL	-(SP)	0916
			67		5E	DD	00104		PUSHL	SP	
			67	0000G	CF	9F	00106		PUSHAB	GET_DEVICE_CONTEXT	
			67		03	FB	0010A		CALLS	#3, -@SYSSCMKRNL	
			67		50	D0	00111		MOVL	R0, STATUS	
			67		54	E8	00114		BLBS	STATUS, 9%	
			67		54	DD	00117		PUSHL	STATUS	0917
			67		01	FB	00119		CALLS	#1, LIB\$STOP	
			67		02	E1	0011C	9%:	BBC	#2, MOUNT_OPTIONS+4, 10%	0919
			67		00	FB	00122		CALLS	#0, MOUNT_DISK2	0920
			67		05	11	00127		BRB	11%	
			67		00	FB	00129	10%:	CALLS	#0, MOUNT_DISK1	0921
			67		01	E1	0012E	11%:	BBC	#1, CLEANOP_FLAGS+1, 12%	0926
			67		8F	E1	00134		BBC	#EXESV REBLDSYSD, EXESGL_STATIC_FLAGS, 12%	0927
			67		8F	DD	00140		PUSHL	#7512091	0930
			67		01	FB	00146		CALLS	#1, LIB\$SIGNAL	
			67		C6	DD	0014D		PUSHL	CHANNEL	0931
			67		01	FB	00151		CALLS	#1, STAND ALONE REBUILD	
			67		C6	D0	00156	12%:	MOVL	LOCK_STATUS+4, R0	0934
			67		11	13	0015B		BEQL	13%	
			67		7E	7C	0015D		CLRQ	-(SP)	0937
			67		7E	D4	0015F		CLRL	-(SP)	
			67		50	DD	00161		PUSHL	R0	
			67		04	FB	00163		CALLS	#4, SYSSDEQ	
			67		C6	D4	0016A		CLRL	LOCK_STATUS+4	0938
			67		01	D0	0016E	13%:	MOVL	#1, R0	0942
			67		04	00	00171		RET		
			67		0000	00	00172	14%:	.WORD	Save nothing	0815

16-Sep-1984 02:12:43 VAX-11 B11sg-32 V4.0-742
14-Sep-1984 13:16:57 [SYSINI.SRC]SYSMOU.B32;1

Page 14
(3)

```

0000V 7E      04      7E  D4 00174      CLRL  -(SP)
      CF      5E      5E  DD 00176      PUSHL SP
      7D 00178      MOVQ  4(AP), -(SP)
      03  FB 0017C      CALLS #3, MAIN_HANDLER
      04 00181      RET

```

; Routine Size: 386 bytes, Routine Base: \$CODES + 0000

32

Syn

EVT
EXE
EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE

EXE
EXE
EXE
EXE
FAI
FAI
FOR
GAI
INI
INI
IOC
IOC
IOC
JOI
LCR
LCR
LCR
LCR
LCR
LCR
LCR

```
.. 415 0943 1 ROUTINE MAIN_HANDLER (SIGNAL, MECHANISM) =
.. 416 0944 1
.. 417 0945 1 ++
.. 418 0946 1
.. 419 0947 1 FUNCTIONAL DESCRIPTION:
.. 420 0948 1
.. 421 0949 1 This routine is the main level condition handler for the MOUNT
.. 422 0950 1 utility. It undoes anything that MOUNT has done so far and then
.. 423 0951 1 unwinds and returns the condition code as status to MOUNT's
.. 424 0952 1 caller (i.e., the CLI).
.. 425 0953 1
.. 426 0954 1
.. 427 0955 1 CALLING SEQUENCE:
.. 428 0956 1 MAIN_HANDLER (ARG1, ARG2)
.. 429 0957 1
.. 430 0958 1 INPUT PARAMETERS:
.. 431 0959 1 ARG1: address of signal array
.. 432 0960 1 ARG2: address of mechanism array
.. 433 0961 1
.. 434 0962 1 IMPLICIT INPUTS:
.. 435 0963 1 NONE
.. 436 0964 1
.. 437 0965 1 OUTPUT PARAMETERS:
.. 438 0966 1 NONE
.. 439 0967 1
.. 440 0968 1 IMPLICIT OUTPUTS:
.. 441 0969 1 NONE
.. 442 0970 1
.. 443 0971 1 ROUTINE VALUE:
.. 444 0972 1 SSS_CONTINUE
.. 445 0973 1
.. 446 0974 1 SIDE EFFECTS:
.. 447 0975 1 stack unwound, control passed to CLI
.. 448 0976 1
.. 449 0977 1 --
.. 450 0978 1
.. 451 0979 2 BEGIN
.. 452 0980 2
.. 453 0981 2 MAP
.. 454 0982 2 SIGNAL : REF BBLOCK, ! signal array
.. 455 0983 2 MECHANISM : REF BBLOCK; ! mechanism array
.. 456 0984 2
.. 457 0985 2 EXTERNAL
.. 458 0986 2 USER_STATUS : VECTOR; ! status return of some routines
.. 459 0987 2
.. 460 0988 2
.. 461 0989 2 ! Do cleanup as indicated by the status flags.
.. 462 0990 2 ! Cause the condition code to be returned in R0 as the main routine value.
.. 463 0991 2
.. 464 0992 2
.. 465 0993 2 IF .SIGNAL[CHFSL_SIG_NAME] NEQ SSS_UNWIND
.. 466 0994 2 AND .BBLOCK[SIGNAL[CHFSL_SIG_NAME], STSSV_SEVERITY] EQL STSSK_SEVERE
.. 467 0995 2 THEN
.. 468 0996 2 BEGIN
.. 469 0997 2
.. 470 0998 2 IF .SIGNAL[CHFSL_SIG_NAME] NEQ 0
.. 471 0999 2 THEN MECHANISM[CRFSL_MCH_SAVRO] = .SIGNAL[CHFSL_SIG_NAME]
```

```

: 472      1000 3      ELSE MECHANISM[CHFSL MCH SAVRO] = .USER_STATUS[0];
: 473      1001 3      IF .LOCK_STATUS [1] REQ 0
: 474      1002 3      THEN
: 475      1003 4          BEGIN
: 476      1004 4              SDEQ (LKID = .LOCK_STATUS [1]);
: 477      1005 4              LOCK_STATUS [1] = 0;
: 478      1006 4          END;
: 479      1007 4          SUNWIND ();
: 480      1008 4      END;
: 481      1009 4
: 482      1010 2      RETURN SS$_CONTINUE;          ! continue from success signals
: 483      1011 2
: 484      1012 1      END;                          ! end of routine MAIN_HANDLER

```

.EXTRN SYSSUNWIND

```

                                0000 00000 MAIN_HANDLER:
                                .WORD      Save nothing
                                MOVL      SIGNAL, R1          : 0943
                                CMPL      4(R1), #2336        : 0993
                                BEQL      4$
                                CMPZV     #0, #3, 4(R1), #4  : 0994
                                BNEQ      4$
                                MOVL      MECHANISM, R0       : 0999
                                TSTL      4(R1)               : 0998
                                BEQL      1$
                                MOVL      4(R1), 12(R0)        : 0999
                                BRB       2$
                                MOVL      USER_STATUS, 12(R0) : 1000
                                MOVL      LOCK_STATUS+4, R0     : 1001
                                BEQL      3$
                                CLRQ      -(SP)               : 1004
                                CLRL      -(SP)
                                PUSHL     R0
                                CALLS     #4, SYSSDEQ
                                CLRL      LOCK_STATUS+4        : 1005
                                CLRQ      -(SP)                : 1007
                                CALLS     #2, SYSSUNWIND
                                MOVL      #1, R0               : 1010
                                RET
                                04 00052

```

: Routine Size: 83 bytes, Routine Base: \$CODE\$ + 0182

```

: 485      1013 1
: 486      1014 1 END
: 487      1015 0 ELUDOM

```

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$GLOBALS	1016	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	80	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	469	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	37	0	1000	00:01.9

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:SYSMOU/OBJ=OBJ\$:SYSMOU MSRC\$:SYSMOU/UPDATE=(ENH\$:SYSMOU)

; Size: 469 code + 1096 data bytes
; Run Time: 00:18.5
; Elapsed Time: 00:36.7
; Lines/CPU Min: 3293
; Lexemes/CPU-Min: 30395
; Memory Used: 169 pages
; Compilation Complete

0390

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY